A. AMENDMENTS TO CLAIMS

Please cancel Claims 5-8, 10-13, 17, 19-22, 27-30, 32-37, 41, 43-44, 53-56, 60, 64 and 68 and amend the claims as indicated hereinafter.

1. (CURRENTLY AMENDED) A method for maintaining data integrity, comprising the computer-implemented steps of:

generating checksum data by a software application performing a physical checksum calculation on a block of data;

after generating said checksum data,

the software application performing a logical check on data contained within the block of data; ~~and~~

one or more components other than the software application performing one or more physical checksum verification procedures prior to writing the block of data to a nonvolatile memory; and

if the block of data passes said logical check, then causing the block of data to be written to the nonvolatile memory.

2. (ORIGINAL) The method of Claim 1 wherein the steps of generating checksum data and performing a logical check are performed in response to a request to write said block of data to nonvolatile memory.

3. (ORIGINAL) The method of Claim 1 further comprising the step of writing the checksum data to nonvolatile memory in association with writing said block of data to nonvolatile memory.

4. (ORIGINAL) The method as recited in Claim 3, further comprising the steps of:

after writing the block of data to nonvolatile memory,

causing the block of data and said checksum data to be read from nonvolatile memory; and

performing a physical checksum verification procedure on said block of data based on said checksum data, wherein the physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

5 – 8. (CANCELED)

9. (CURRENTLY AMENDED) A method for maintaining data integrity, comprising the computer-implemented steps of:

generating checksum data by performing a physical checksum calculation on a block of data;

after generating the checksum data,

performing a logical check on data contained within the block of data;

if the block of data passes the logical check, then

causing the block of data to be written to nonvolatile memory; and

writing the checksum data to the nonvolatile memory in association with writing the block of data to the nonvolatile memory; and

after writing the block of data to the nonvolatile memory,

causing the block of data and the checksum data to be read from the nonvolatile memory; and

performing a physical checksum verification procedure on the block of data based on the checksum data, wherein the physical checksum verification

procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

~~The method as recited in Claim 4, further comprising the step of:~~

~~after performing the physical checksum verification procedure on said block of data,~~

~~storing the block of data as a backup version of the block of data, wherein the backup version of the block of data is maintained separate from said block of data in said nonvolatile memory.~~

10 - 13. (CANCELED)

14. (PREVIOUSLY PRESENTED) A method for maintaining data integrity, comprising the computer-implemented steps of:

a software application performing a physical checksum calculation on a block of data;

after performing the physical checksum calculation,

a component other than the software application performing a first physical checksum verification procedure on said block of data prior to writing the block of data to a nonvolatile memory, wherein the first physical checksum verification procedure indicates whether the block of data was corrupted subsequent to the software application performing the physical checksum calculation on the block of data; and

if the block of data passes said first physical checksum verification procedure, then causing the block of data to be written to the nonvolatile memory.

15. (PREVIOUSLY PRESENTED) The method as recited in Claim 14, further comprising the steps of:

after writing the block of data to the nonvolatile memory,

causing the block of data to be read from the nonvolatile memory; and

performing a second physical checksum verification procedure on said block of data, wherein the second physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the first physical checksum verification procedure on the block of data.

16. (PREVIOUSLY PRESENTED) The method as recited in Claim 14, further comprising performing one or more other physical checksum verification procedures on said block of data prior to writing the block of data to the nonvolatile memory, wherein the one or more other physical checksum verification procedures indicate whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data.

17. (CANCELED)

18. (PREVIOUSLY PRESENTED) The method as recited in Claim 14, wherein the step of a component other than the software application performing a first physical checksum verification procedure on said block of data comprises the step of a disk array component performing the first physical checksum verification procedure on said block of data, wherein the disk array component is configured to write the block of data to disk only after verifying the integrity of the data block.

19 - 22. (CANCELED).

23. (CURRENTLY AMENDED) A computer-readable medium for maintaining data integrity, the computer-readable medium carrying one or more sequences of one or more instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of:

generating checksum data by a software application performing a physical checksum calculation on a block of data;

after generating said checksum data,

the software application performing a logical check on data contained within the block of data; ~~and~~

one or more components other than the software application performing one or more physical checksum verification procedures prior to writing the block of data to a nonvolatile memory; and

if the block of data passes said logical check, then causing the block of data to be written to the nonvolatile memory.

24. (PREVIOUSLY PRESENTED) The computer-readable medium of Claim 23 wherein the steps of generating checksum data and performing a logical check are performed in response to a request to write said block of data to nonvolatile memory.

25. (PREVIOUSLY PRESENTED) The computer-readable medium of Claim 23 further comprising one or more sequences of additional instructions which, when executed by the one or more processors, cause the one or more processors to perform the step of writing the checksum data to nonvolatile memory in association with writing said block of data to nonvolatile memory.

26. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in Claim 25, further comprising one or more sequences of additional instructions which, when executed by the one or more processors, cause the one or more processors to perform the steps of:

after writing the block of data to nonvolatile memory,

causing the block of data and said checksum data to be read from nonvolatile memory; and

performing a physical checksum verification procedure on said block of data based on said checksum data, wherein the physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

27 - 30. (CANCELED)

31. (CURRENTLY AMENDED) A computer-readable medium for maintaining data integrity, the computer-readable medium carrying instructions which, when processed by one or processors, cause:

generating checksum data by performing a physical checksum calculation on a block of data;

after generating the checksum data,

performing a logical check on data contained within the block of data;

if the block of data passes the logical check, then

causing the block of data to be written to nonvolatile memory; and

writing the checksum data to the nonvolatile memory in association with writing the block of data to the nonvolatile memory; and

after writing the block of data to the nonvolatile memory,

causing the block of data and the checksum data to be read from the nonvolatile memory; and

performing a physical checksum verification procedure on the block of data based on the checksum data, wherein the physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

~~The computer-readable medium as recited in Claim 26, further comprising one or more sequences of additional instructions which, when executed by the one or more processors, cause the one or more processors to perform the step of:~~

~~after performing the physical checksum verification procedure on said block of data, storing the block of data as a backup version of the block of data, wherein the backup version of the block of data is maintained separate from said block of data in said nonvolatile memory.~~

32 - 37. (CANCELED)

38. (PREVIOUSLY PRESENTED) A computer-readable medium for storing data in a nonvolatile memory, the computer-readable medium carrying one or more sequences of one or more instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of:

a software application performing a physical checksum calculation on a block of data;

after performing the physical checksum calculation,

a component other than the software application performing a first physical checksum verification procedure on said block of data prior to writing the block of data to a nonvolatile memory, wherein the first physical checksum verification procedure indicates whether the block of data was corrupted subsequent to the software application performing the physical checksum calculation on the block of data; and

if the block of data passes said first physical checksum verification procedure, then causing the block of data to be written to the nonvolatile memory.

39. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in Claim 38, further comprising one or more sequences of additional instructions which, when executed by the one or more processors, cause the one or more processors to perform the steps of:

after writing the block of data to the nonvolatile memory,

causing the block of data to be read from the nonvolatile memory; and

performing a second physical checksum verification procedure on said block of data, wherein the second physical checksum verification procedure indicates whether the block of data was corrupted subsequent to performing the first physical checksum verification procedure on the block of data.

40. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in Claim 38, further comprising performing one or more other physical checksum verification procedures on said block of data prior to writing the block of data to the nonvolatile memory, wherein the one or more other physical checksum verification procedures

indicate whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data.

41. (CANCELED)

42. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in ~~Claim 41,~~ Claim 38, wherein the step of a component other than the software application performing a first physical checksum verification procedure on said block of data comprises the step of a disk array component performing the first physical checksum verification procedure on said block of data, wherein the disk array component is configured to write the block of data to disk only after verifying the integrity of the data block.

43 - 44. (CANCELED)

45. (PREVIOUSLY PRESENTED) A storage device comprising:

a software application configured to perform a physical checksum calculation on a block of data; and

a component configured to

perform a first physical checksum verification procedure on the block of data prior to writing the block of data to a nonvolatile memory, wherein the first physical checksum verification procedure indicates whether the block of data was corrupted subsequent to the software application performing the physical checksum calculation on the block of data, and

if the block of data passes said first physical checksum verification procedure, then cause the block of data to be written to the nonvolatile memory.

46. (PREVIOUSLY PRESENTED) The storage device as recited in Claim 45, wherein the component is further configured to after writing the block of data to the nonvolatile memory,

cause the block of data to be read from the nonvolatile memory; and

perform a second physical checksum verification procedure on said block of data,

wherein the second physical checksum verification procedure indicates whether the block of data was corrupted subsequent to the component performing the first physical checksum verification procedure on the block of data.

47. (CURRENTLY AMENDED) The storage device as recited in ~~Claim 46,~~ Claim 45, wherein the component is further configured to perform one or more other physical checksum verification procedures on said block of data prior to writing the block of data to the nonvolatile memory, wherein the one or more other physical checksum verification procedures indicate whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data.

48. (PREVIOUSLY PRESENTED) The storage device as recited in Claim 45, wherein the component is a disk array.

49. (CURRENTLY AMENDED) A storage device comprising:

a storage medium; ~~and~~

~~a storage mechanism communicatively coupled to the storage medium, the storage mechanism being configured to:~~

a software application configured to perform a logical check on data contained in a block of data after the software application has previously performed a physical

checksum calculation ~~has previously been performed~~ on the block of data; and ~~data, and~~

one or more components other than the software application configured to perform one or more physical checksum verifications prior to allowing the block of data to be written to the storage medium, wherein the one or more physical checksum verifications indicate whether the block of data was corrupted subsequent to the physical checksum calculation;

wherein the software application is further configured to allow the block of data to be written to the storage medium if the block of data passes the logical check. ~~check, then allowing the block of data to be written to the storage medium.~~

50. (PREVIOUSLY PRESENTED) The apparatus as recited in Claim 49, wherein the physical checksum calculation and the logical check are both performed in response to a request to write the block of data to the storage medium.

51. (CURRENTLY AMENDED) The apparatus as recited in Claim 49, wherein the storage ~~mechanism~~ device is further configured to write, to the storage medium, checksum data generated by the physical checksum calculation.

52. (CURRENTLY AMENDED) The apparatus as recited in Claim 51, wherein the storage ~~mechanism~~ device is further configured to

after the block of data is written to the storage medium,

cause the block of data and the checksum data to be read from the storage medium; and

perform a physical checksum verification on the block of data based on the checksum data, wherein the physical checksum verification indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

53 - 56. (CANCELED)

57. (CURRENTLY AMENDED) A storage device comprising:

a storage medium; and

a storage mechanism communicatively coupled to the storage medium and being configured to

generate checksum data by performing a physical checksum calculation on a block of-data;

after generating the checksum data,

perform a logical check on data contained within the block of data;

if the block of data passes the logical check, then

cause the block of data to be written to nonvolatile memory; and

write the checksum data to the nonvolatile memory in association with writing the block of data to the nonvolatile memory; and

after writing the block of data to the nonvolatile memory,

cause the block of data and the checksum data to be read from the nonvolatile memory; and

perform a physical checksum verification procedure on the block of data based on the checksum data, wherein the physical checksum

verification procedure indicates whether the block of data was corrupted subsequent to performing the logical check on the data contained with the block of data.

~~The apparatus as recited in Claim 52, wherein the storage mechanism is further configured to after performing the physical checksum verification on the block of data, causing the block of data to be stored on the storage medium as a backup version of the block of data, wherein the backup version of the block of data is maintained separate from the block of data on the storage medium.~~

58. (CURRENTLY AMENDED) A method for maintaining data integrity, the method comprising the computer-implemented steps of:

a software application performing a physical checksum calculation on a block of data;

a software application performing a logical check on the block of data;

one or more components other than the software application performing a physical checksum verification on the block of data, wherein the physical checksum verification indicates whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data; and

if the block of data passes both the logical check and the physical checksum verification, then allowing the block of data to be written to a nonvolatile memory.

59. (PREVIOUSLY PRESENTED) The method as recited in Claim 58, wherein the logical check is performed prior to the performance of the physical checksum verification.

60. (CANCELED)

61. (PREVIOUSLY PRESENTED) The method as recited in Claim 58, wherein the logical check is performed after the physical checksum calculation.

62. (CURRENTLY AMENDED) A computer-readable medium for maintaining data integrity, the computer-readable medium carrying one or more sequences of instructions which, when executed by one or more processors, cause performance of the steps of:

a software application performing a physical checksum calculation on a block of data;

a software application performing a logical check on the block of data;

one or more components other than the software application performing a physical checksum verification on the block of data, wherein the physical checksum verification indicates whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data; and

if the block of data passes both the logical check and the physical checksum verification, then allowing the block of data to be written to a nonvolatile memory.

63. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in Claim 62, wherein the logical check is performed prior to the performance of the physical checksum verification.

64. (CANCELED)

65. (PREVIOUSLY PRESENTED) The computer-readable medium as recited in Claim 62, wherein the logical check is performed after the physical checksum calculation.

66. (CURRENTLY AMENDED) An apparatus for maintaining data integrity, the apparatus comprising a memory storing instructions which, when executed by one or more processors, cause performance of the steps of:

a software application performing a physical checksum calculation on a block of data;

a software application performing a logical check on the block of data;

one or more components other than the software application performing a physical checksum verification on the block of data, wherein the physical checksum verification indicates whether the block of data was corrupted subsequent to performing the physical checksum calculation on the block of data; and

if the block of data passes both the logical check and the physical checksum verification, then allowing the block of data to be written to a nonvolatile memory.

67. (PREVIOUSLY PRESENTED) The apparatus as recited in Claim 66, wherein the logical check is performed prior to the performance of the physical checksum verification.

68. (CANCELED)

69. (PREVIOUSLY PRESENTED) The apparatus as recited in Claim 66, wherein the logical check is performed after the physical checksum calculation.